Empirical Measurements of Service Discovery Technologies

Olivier Mathieu, Doug Montgomery, Scott Rose

NIST Information Technology Laboratory
Advanced Networking Technologies Division

Networking for Pervasive Computing Project
proj-netpc@antd.nist.gov

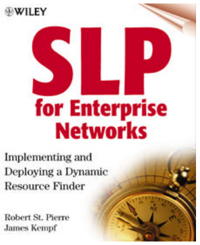
Performance and Scaling of Service Discovery Protocols

- **What are the performance requirements for SDPs?**
  - 10s of geriactic patients sitting / moving slowly?
  - 100s-1000s of shoppers walking about?
  - 10,000s-? cars on a highway?
- **Scale**
  - Number of nodes, services, directories, and clients?
  - Size of PC network topologies?
- **Dynamics**
  - Rate of service / client arrival departure?
  - Service load of advertisements, queries, control, events?
  - Latency requirements for service discovery?
- **Network Technologies**
  - Range of link technologies?
  - Performance of WAN connections?

SDP Performance / Scalability Measurements

Objective: Methodologies and tools for comparative performance and scaling analysis of live SDP implementations.

Initial focus: JINI Universal Plug and Play


WILEY
SLP for Enterprise Networks
Implementing and Deploying a Dynamic Resource Finder
Robert St. Pierre
James Kempf

Approach:

- Design of technology independent **benchmark service / scenarios**.
- Development of **synthetic workload generation tools** for emulating the behavior of large scale dynamic ad hoc networking environments.
- Development of implementation independent **performance measurement methodologies and tools** for SDPs and supporting protocols.

Performance Measurement Methodologies

- SDP Architectures / Protocols Considerably Different
  - Directory based vs flat peer-to-peer (combinations)
  - Java RMI and Serialized Objects vs HTTP / SOAP / GENA and XML
- Usage Based Scenarios & Metrics
  - Service initiation – auto configuration, advertisement, renewal
  - Client type query – single instance, multiple instances, all instances
  - Client instance query – query for existing service, persistent query
  - Client event notification – registration latency, notification latency, distributed control performance (control + eventing).
- Implementation Independent Measurement Tools.
  - Measure on-the-wire
  - Response / Load

SDP Independent Benchmark Service

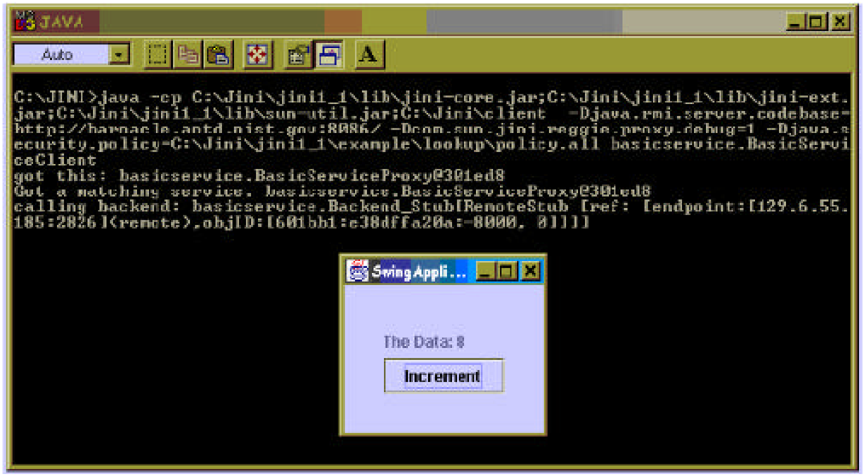
- **Objective** – workload basis for meaningful comparative comparisons of Jini / UPnP performance.
  - – Simple device / service that can be used to exercise all significant discovery / control capabilities of Jini and UPnP.
- **Benchmark Service** – very simple counting device.
  - – Capabilities - Get / Set integer counter.
  - – Attributes – GID, Name, Type
    - Enable multiple match / query semantics
  - – Service interfaces
    - Control – get / set integer
    - GUI – simple user interface for control
    - Eventing – remote notification of counter change
- Jini and UPnP instantiations

Jini Benchmark Service

```
/*
 * BasicService Interface
 * This is the interface for the Basic Jini service for
 * the client side.
 *
 * Scott Rose
 * NIST
 * 9/6/00
 */
package basicservice;

import java.rmi.RemoteException;
import net.jini.core.event.RemoteEventListener;
import net.jini.core.event.EventRegistration;

public interface BasicServiceIF
{
   public int getData() throws RemoteException;
   public void setData(int newVal) throws RemoteException;
   public EventRegistration addRemoteListener(RemoteEventListener rev)
                  throws RemoteException;
   public void getGUI() throws RemoteException;
}
```



```
C:\JINI>java -cp C:\Jini\jini1_1\lib\jini-core.jar;C:\Jini\jini1_1\lib\jini-ext.
jar;C:\Jini\jini1_1\lib\sun-util.jar;C:\Jini\client  -Djava.rmi.server.codebase=
http://barnacle.antd.nist.gov:8086/ -Dcom.sun.jini.reggie.proxy.debug=1 -Djava.s
ecurity.policy=C:\Jini\jini1_1\example\lookup\policy.all basicservice.BasicServi
ceClient
got this: basicservice.BasicServiceProxy@301ed8
Got a matching service. basicservice.BasicServiceProxy@301ed8
calling backend: basicservice.Backend_Stub[RemoteStub [ref: [endpoint:[129.6.55.
185:2826](remote),objID:[601bb1:e38dffa20a:-8000, 0]]]]
```

The Data: 8

UPnP Benchmark Service

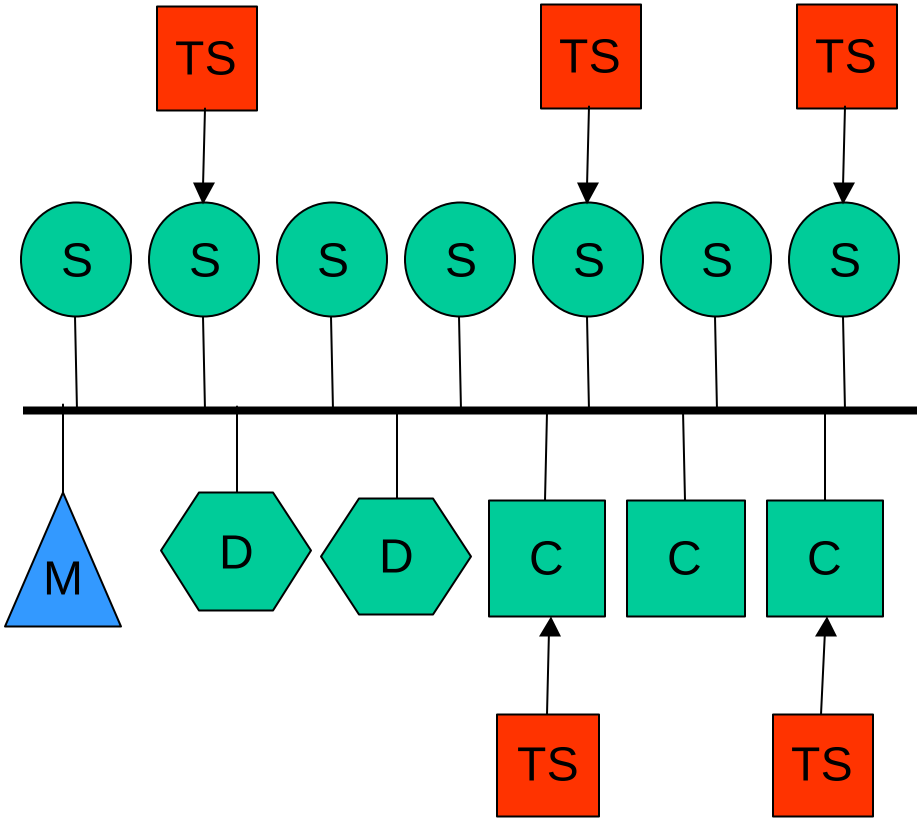
```
<?xml version="1.0"?>
<root xmlns="urn:schemas-upnp-org:device-1-0">
<URLBase>http://129.6.51.81:20002</URLBase>
<device>
<deviceType>urn:schemas-upnp-org:device:basicdevice:1</deviceType>
        <friendlyName>Basic Service for Service Discovery Protocol
Testing</friendlyName>
        <manufacturer>NIST-ANTD-ITG</manufacturer>
        <manufacturerURL>http://w3.antd.nist.gov</manufacturerURL>
        <modelDescription>UPnP Basic Service 1.0</modelDescription>
        <modelName>BasicService</modelName> <modelNumber>1.0</modelNumber>
        <modelURL>http://w3.antd.nist.gov/modelURL</modelURL>
        <serialNumber>123456789001</serialNumber> <UDN>uuid:Upnp-BasicService-1_0-
darwin-20002</UDN>
        <UPC>123456789</UPC>
        <serviceList>
            <service>
                <serviceType>urn:schemas-upnp-
org:service:basicservice:1</serviceType>
                <serviceId>urn:upnp-org:serviceId:basicservice1</serviceId>
                <controlURL>/upnp/control/basicservice1</controlURL>
                <eventSubURL>/upnp/event/basicservice1</eventSubURL>
                <SCPDURL>/basicserviceSCPD.xml</SCPDURL>
            </service>
        </serviceList>
        <presentationURL>/basicdevice.html</presentationURL>
    </device>
</root>
```

Synthetic Workload Generation Tools

- **Objective** – Emulate large, dynamic environments of 100's of devices / services and 10's of control points / clients.
  - – Dynamic devices providing the benchmark service.
  - – Scripted control points execute measurement scenarios.
- Jini and UPnP **Experimenters Toolkits**
  - – Drive real implementations: SunMS Jini, Intel Linux & Windows ME UPnP.
  - – Emulate the behavior of a large number of dynamic devices
    - # devices, device creation rate, device life time, service life time
    - Devices implement the benchmark service
  - – Emulate the behavior control points / scripted behavior for testing
    - # clients, query workload – (query type, service names / types)
- Jini / UPnP **Device Emulation Tools**
  - – Initial development complete – target of 100's devices and 10's of control points met.
  - – Discovered scaling issues in Linux UPnP 1.x SDK

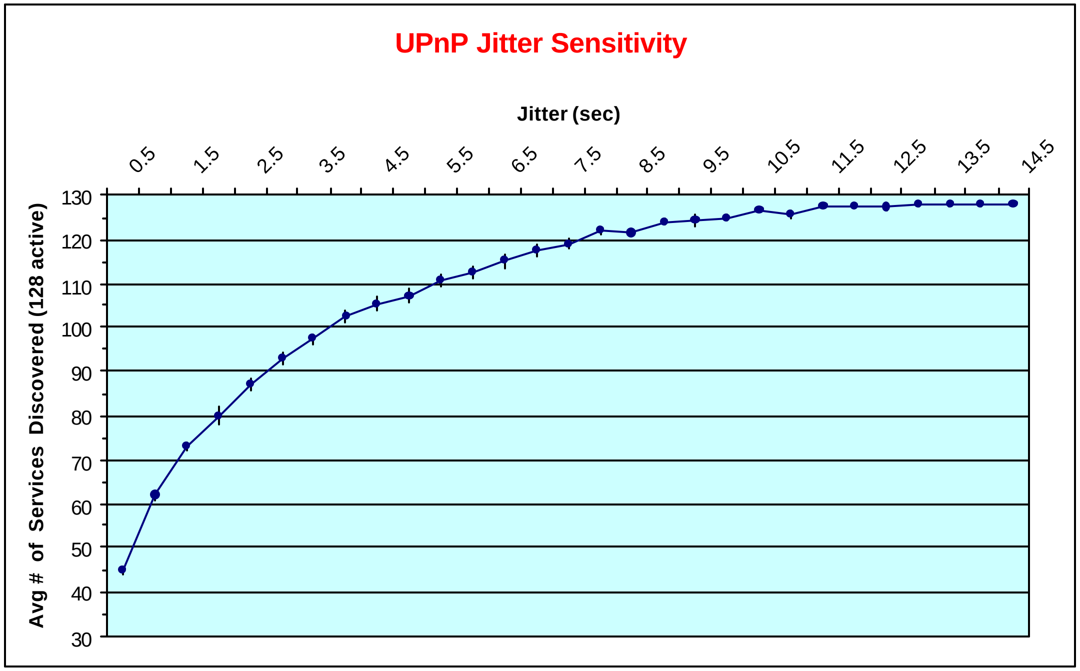
Emulated PC Networking Environments

- Emulated SDP Components
  - **S** - services
  - **C** - clients
  - **D** – directories
  - **M** – measurement points
- Device Dynamics
  - Birth / death processes
- Scripted Behavior
  - **TS** – Test Scripts



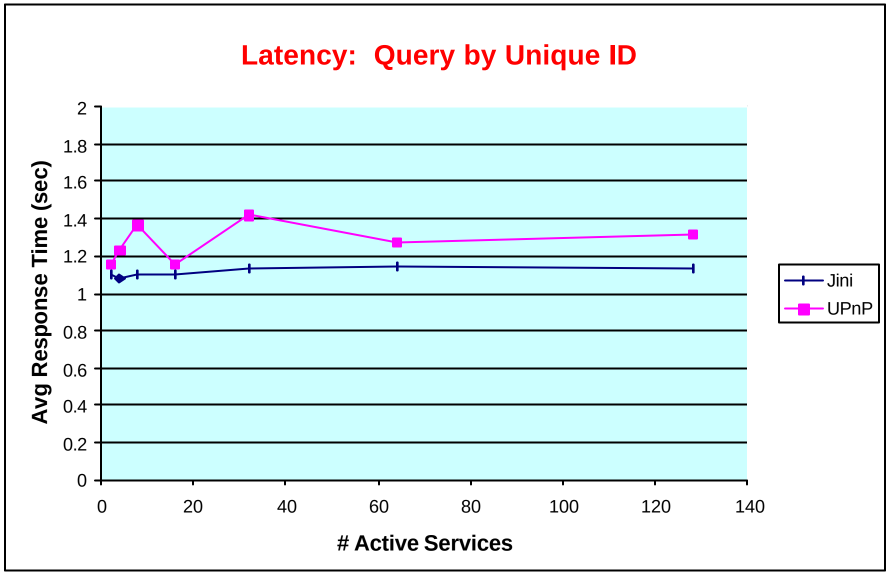
Linux UPnP Scaling Problems

- Problems encountered in achieving initial scaling goals for device emulation tools.
  - UPnP scalability above 40 devices a function of protocol tuning parameters (e.g., response jitter, multicast retransmission factor).
  - Errors in implementation of jitter algorithms

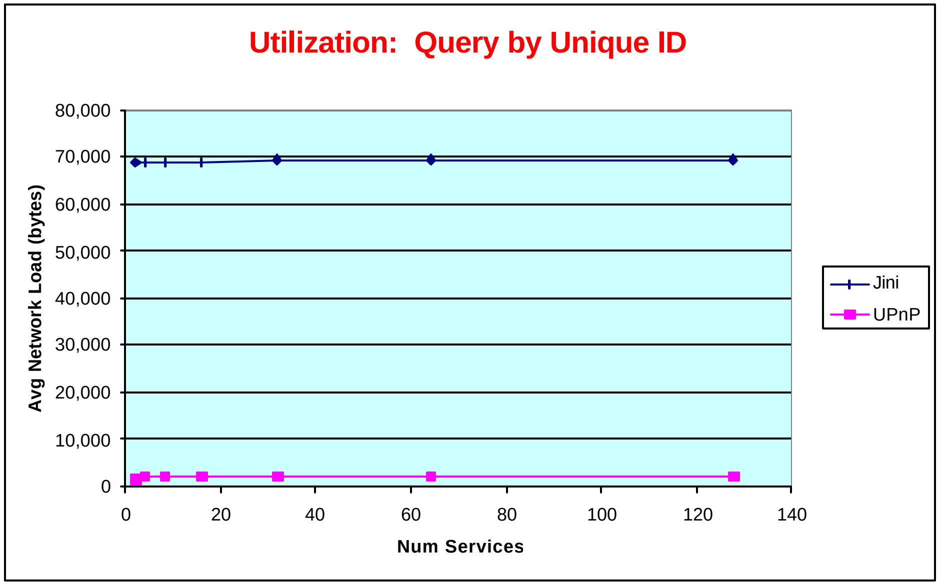


Some Example Results: Jini vs UPnP Discovery

Initial experiments to establish UPnP / Jini baseline performance.



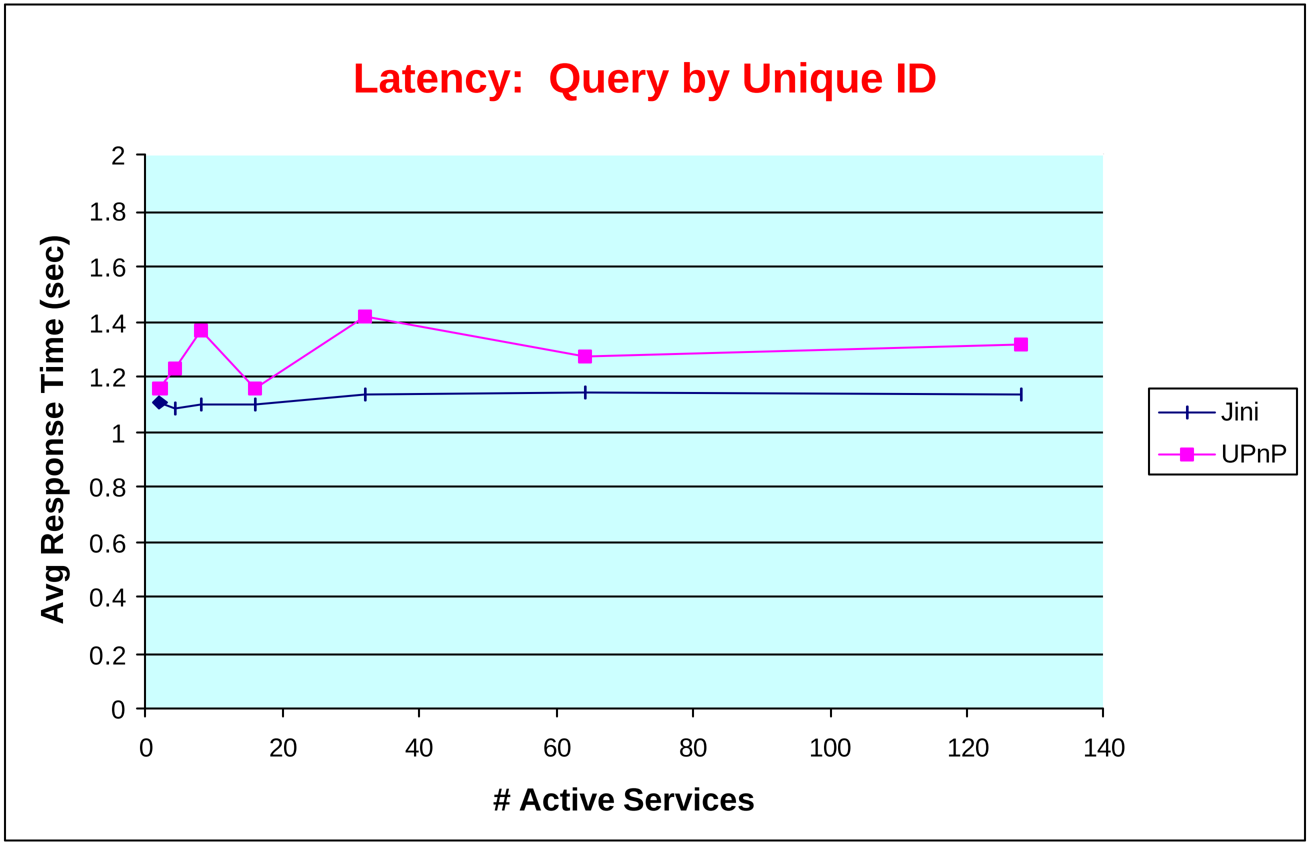
- **Latency**: Time from query until full response returned.



- **Utilization**: Query / response transport data.

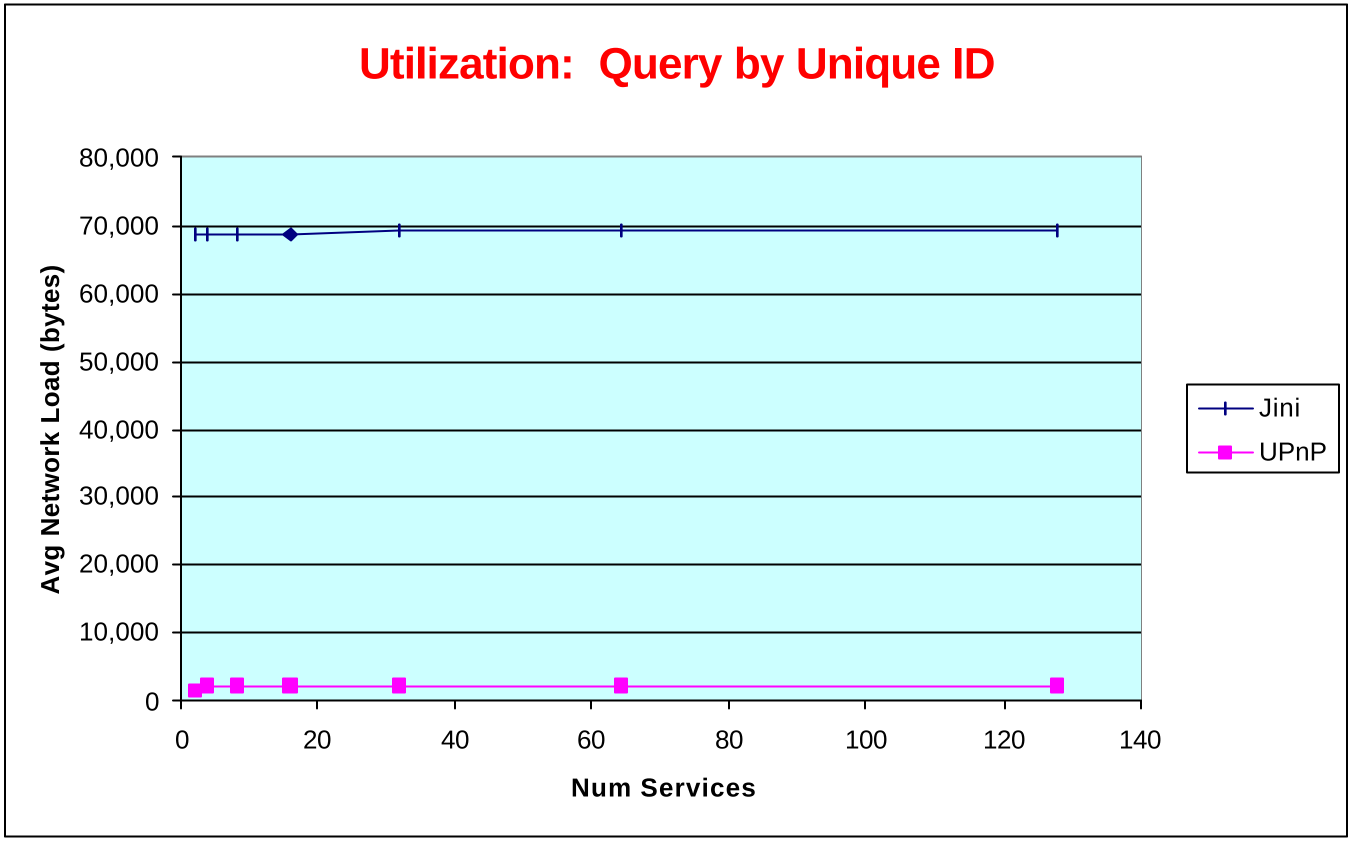
Some Example Results: Jini vs UPnP Discovery

- **Initial experiments to establish UPnP / Jini baseline performance**



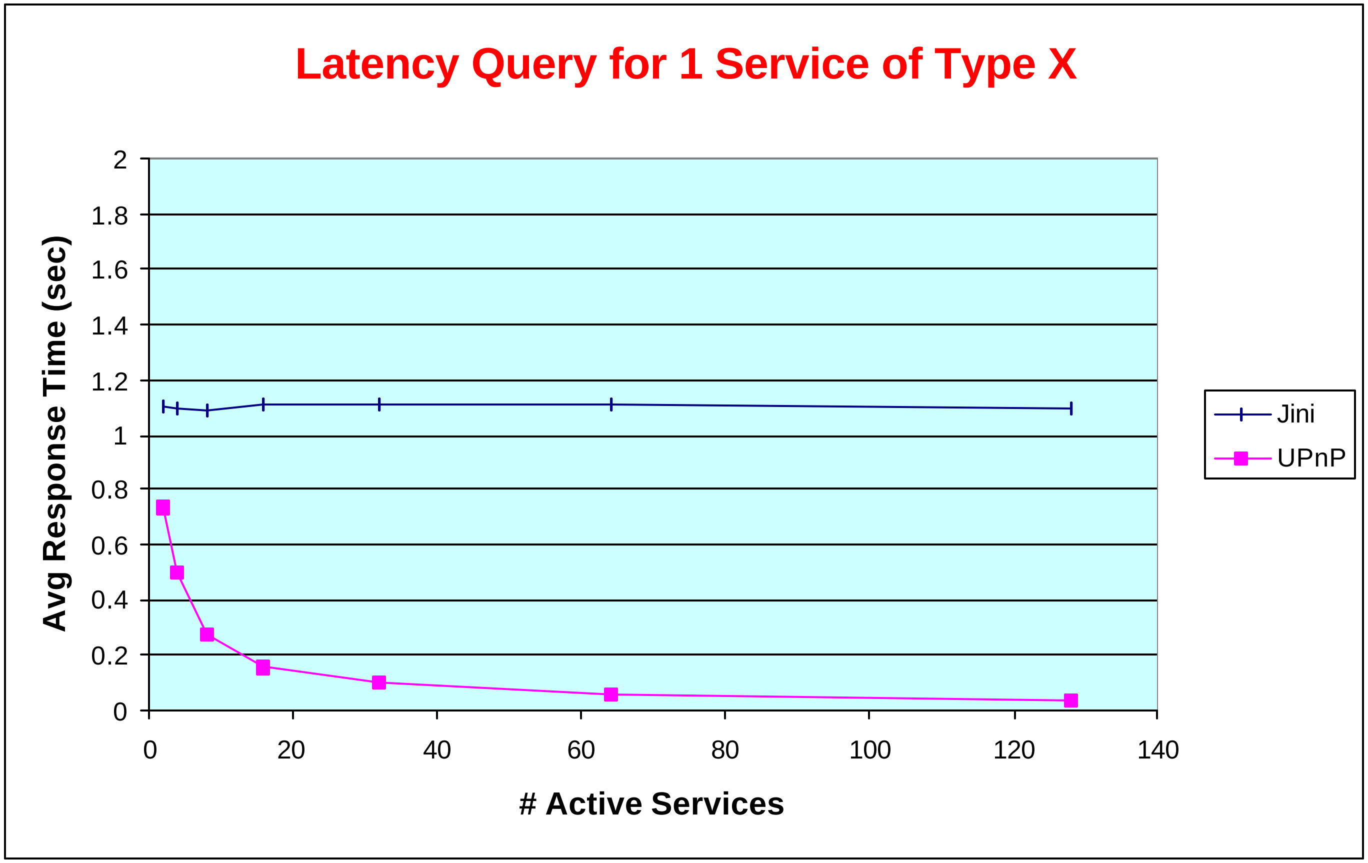
Some Example Results: Jini vs UPnP Discovery

- Initial experiments to establish UPnP / Jini baseline performance



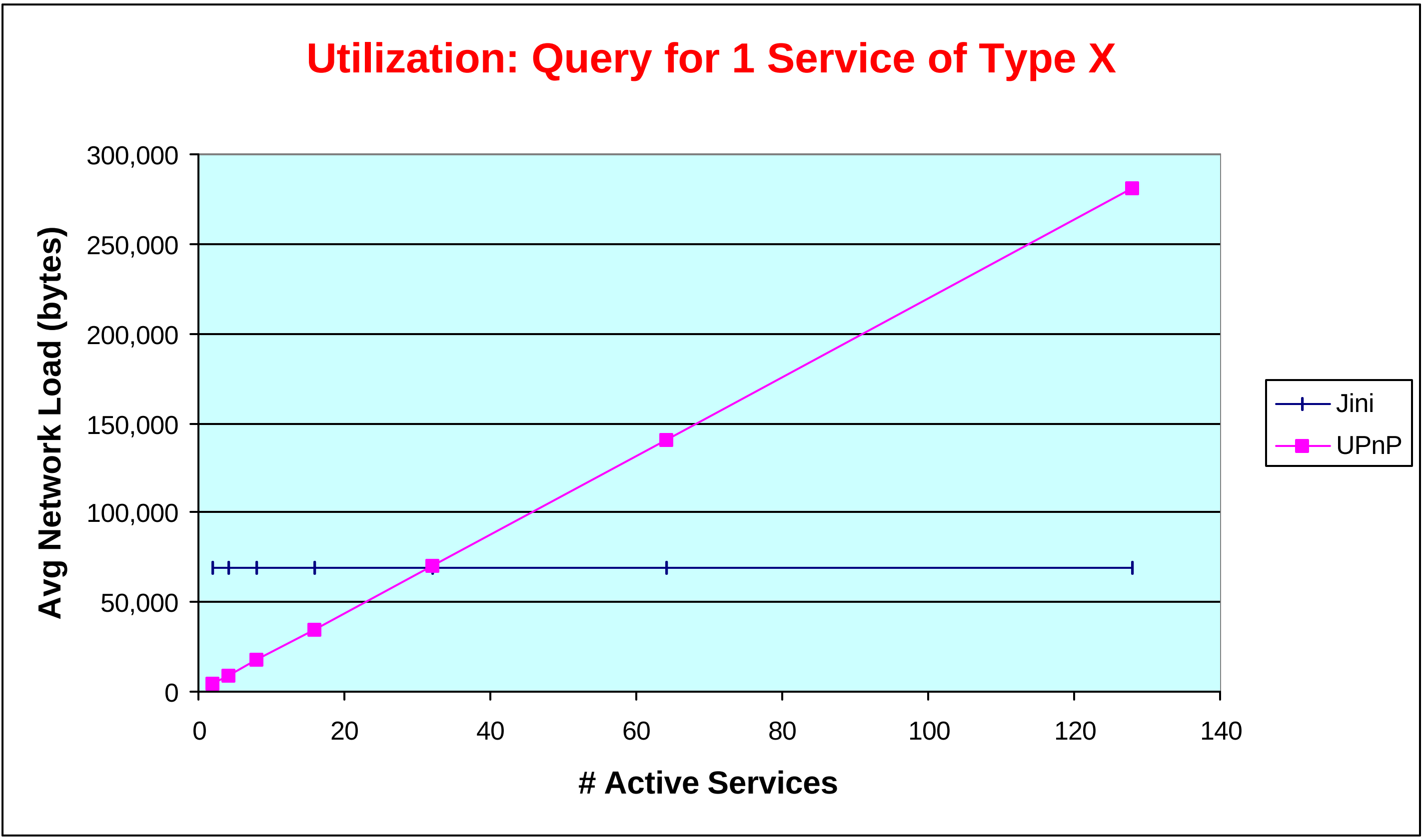
Some Example Results: Jini vs UPnP Discovery

- Performance of “anycast” Query: “Find one instance of type X”



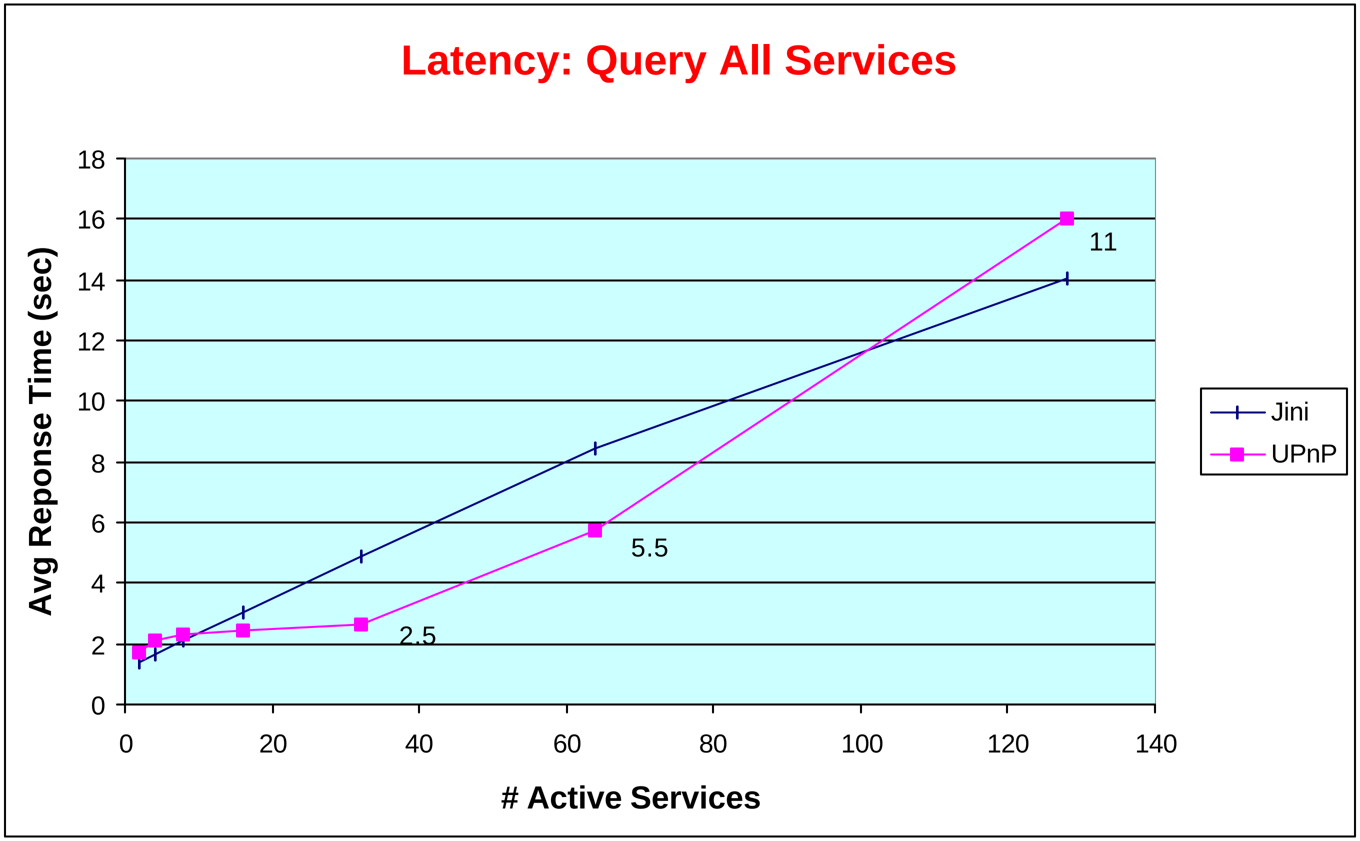
Some Example Results: Jini vs UPnP Discovery

- Performance of “anycast” Query: “Find one instance of type X”



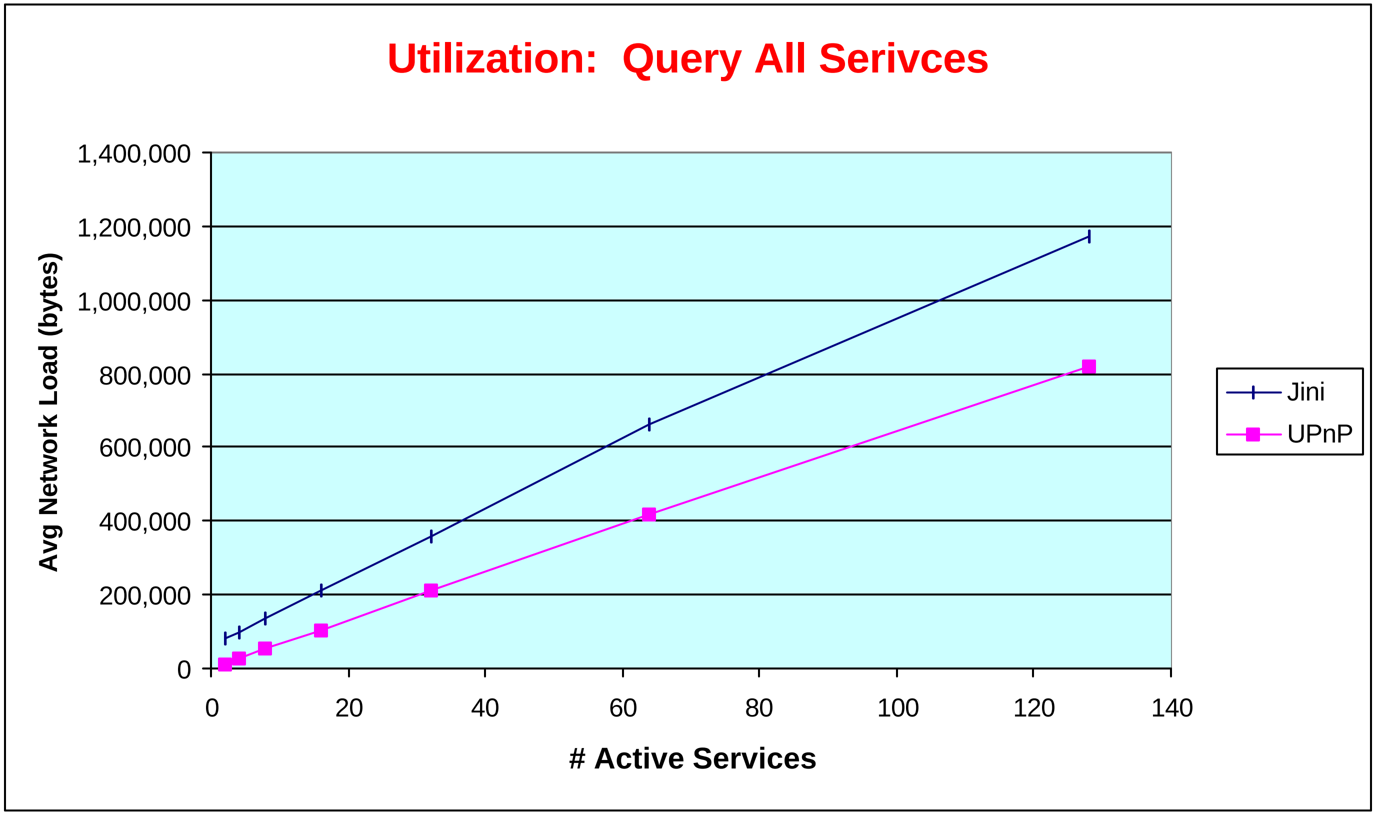
Some Example Results: Jini vs UPnP Discovery

- Performance of device poll: “Find all active devices / services”.



Some Example Results: Jini vs UPnP Discovery

- Performance of device poll: “Find all active devices / services”.



Status of SDP Performance and Scalability Effort

Acomplishments:

- Methodology and scenarios for comparative evaluations of SDPs
- Synthetic workload generation / emulation tools for Jini and UPnP.
- Performance measurement tools for SDPs and supporting protocols.
- Initial studies of client query scenarios.

Near Term Plans:

- Expand testbed with 802.11 and NIST Net emulated WAN links
- Complete eventing studies
- Incorporate SLP
- Develop simulation framework for more advanced study of SDP behavior

Any Questions?

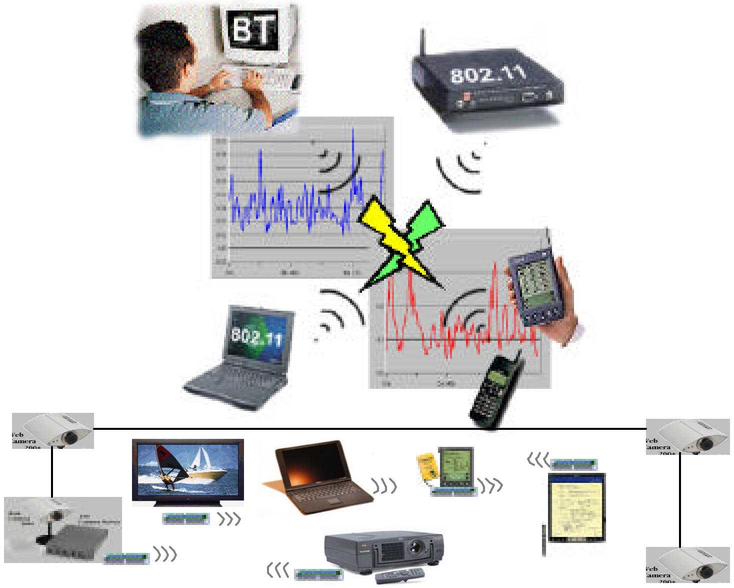
..or Would you rather run out and join the DC beltway at rush hour?

NIST National Institute of Standards and Technology Technology Administration, U.S. Department of Commerce

NIST CENTENNIAL 1901-2001

DARPA ARDA

Measuring SDP Performance and Scalability



Objectives

(1) Provide a quantitative, comparative analysis of the performance and scaling characteristics of emerging service discovery protocols (SDPs).
(3) Design methodologies and tools for performance and scaling measurement of SDPs and supporting protocols.
(4) Develop simulation tools for large scale ad-hoc network / application environments

Technical Approach

- Design and develop experimenters toolkits for conducting live performance analysis of SPDs implementations.
- Propose metrics and scenarios for comparing the performance of multiple SDP protocols.
- Design and develop simulation models of emerging SDPs and adhoc network environments.
- Analyze and compare the performance of SDPs based upon testbed measurements and simulation.

Recent Accomplishments:

- Designed methodology and scenarios for comparative performance evaluation of live Jini and UPnP implementations.
- Established testbed with Jini, UPnP implementations.
- Developed synthetic workload generation tools for Jini and UPnP capable of emulating 10's-100's of devices/services and control point / clients.
- Discovered scaling problems with Linux UPnP 1.0 implementation. Conducted initial investigations in protocol / parameter tuning to increase the scalability of this implementation.
- Began design and development of on-the-wire performance measurement tools for SDPs and supporting protocols.

Products & Contributions

- Experimenter's toolkits consisting of synthetic workload generation tools, scenario scripts, and performance measurement tools for SDPs.
- Measurement methodologies and tools for SDPs and supporting protocols.
- Ad-hoc network simulation environment and SDP protocol models.
- Publications / standards contributions providing quantitative analysis of the relative performance and scaling properties of SDPs.